The Claims

Claims 1-18. (Canceled).

19. (Original) A method comprising:

receiving an identifier;

generating, based on the identifier, a mapped identifier;

encoding the mapped identifier; and

encrypting the encoded identifier.

20. (Original) A method as recited in claim 19, wherein the identifier comprises one of: a file name, a folder name, and a directory name.

21. (Original) A method as recited in claim 19, further comprising:

generating, based on the mapped identifier, a decasified identifier and corresponding case information;

wherein the encoding comprises encoding the decasified identifier; and

wherein the encrypting comprises encrypting both the encoded decasified identifier and the case information.

22. (Original) A method as recited in claim 21, wherein generating the decasified identifier and corresponding case information comprises:

for each character that has both an upper-case and a lower-case form, storing the character in upper-case form and recording in the case information whether the character was in upper-case form or lower-case form.

23. (Original) A method as recited in claim 22, further comprising:

storing the character in upper-case form only if the character is one of particular set of characters; and

storing the character without altering its case if the character is not one of the particular set of characters.

24. (Original) A method as recited in claim 23, wherein the particular set of characters comprises the extended ASCII character set.

25. (Original) A method as recited in claim 19, wherein the generating comprises generating the mapped identifier only if the received identifier is syntactically legal.

26. (Original) A method as recited in claim 19, wherein the encoding comprises encoding the mapped identifier only if the received identifier is syntactically legal.

27. (Canceled).

28. (Canceled).

29. (Original) A method as recited in claim 19, wherein generating the mapped identifier comprises:

checking whether the identifier is equal to one of a plurality of illegal identifiers;

if the identifier is not equal to one of the plurality of illegal identifiers, then checking whether the identifier is equal to one of the plurality of illegal identifiers followed by one or more particular characters;

if the identifier is not equal to one of the plurality of illegal identifiers followed by one or more particular characters, then using the identifier as the mapped identifier; and

if the identifier is equal to one of the plurality of illegal identifiers followed by one or more particular characters, then using as the mapped identifier the identifier with one of the particular characters removed.

30. (Original) A method as recited in claim 29, wherein the particular character comprises an underscore.

31. (Original) A method as recited in claim 19, wherein encoding the mapped identifier comprises:

reversing the order of characters in the mapped identifier;

removing, from the reversed identifier, all trailing characters of a particular type;

initializing the encoded identifier with a string of one bits equal in number to a number of trailing characters removed form the reversed identifier followed by a zero bit;

selecting a first character from the reversed identifier;

encoding the first character using a first coding table;

adding, to the encoded identifier, a series of zero bits followed by the encoded first character;

for each additional character in the reversed identifier,

selecting the next character in the reversed identifier,

encoding the next character using a second coding table,

adding, to the encoded identifier, a series of zero bits followed by the encoded next character; and

removing any trailing zero bits and the one bit preceding the trailing zero bits from the encoded identifier.

32. (Original) A method as recited in claim 31, wherein the characters of a particular type are the characters that are coded to zero using the first coding table.

33. (Original) A method as recited in claim 31, wherein the first coding table and the second coding table are Huffman coding tables.

34. (Original) A method as recited in claim 31, wherein each coding in the first coding table is the same as a corresponding coding in the second coding table, but the second coding table codes additional characters not coded by the first coding table.

35. (Original) A method as recited in claim 31, wherein for the first character and each additional character, encoding the character only if a set of leading bits of the character are zero, and further comprising adding the character to the encoded identifier if the set of leading bits of the character are not zero.

36. (Original) A method as recited in claim 19, wherein encoding the mapped identifier comprises:

reversing the order of characters in the mapped identifier;

removing, from the reversed identifier, all trailing characters of a particular type;

initializing the encoded identifier with a string of one bits equal in number to a number of trailing characters removed form the reversed identifier followed by a zero bit;

selecting a first character from the reversed identifier;

encoding the first character using a first coding table;

adding, to the encoded identifier, a series of zero bits followed by the encoded first character;

for each additional character in the reversed identifier,

selecting the next character in the reversed identifier,

encoding the next character using one of a plurality of additional coding tables,

adding, to the encoded identifier, a series of zero bits followed by the encoded next character; and

removing any trailing zero bits and the one bit preceding the trailing zero bits from the encoded identifier.

37. (Original) A method as recited in claim 19, wherein encrypting the encoded identifier comprises using a block cipher to encrypt the encoded identifier.

38. (Original) A system as recited in claim 19, wherein encrypting the encoded identifier comprises using cipher block chaining to encrypt the encoded identifier.

39. (Original) A system as recited in claim 19, wherein the encrypting comprises encrypting the encoded identifier to generate, using a block cipher, a ciphertext having a fixed size.

40. (Original) A system as recited in claim 39, further comprising indicating that the received identifier cannot be encrypted if the length of the encoded identifier exceeds the fixed size by more than one.

41. (Original) One or more computer-readable memories containing a computer program that is executable by a processor to perform the method recited in claim 19.

Claims 42-46 (Canceled).

47. (Original) One or more computer-readable media having stored thereon a plurality of instructions that, when executed by one or more processors of a computer, causes the one or more processors to perform acts including:

receiving a plaintext identifier;

generating a ciphertext by encrypting the plaintext identifier only if the plaintext identifier is syntactically legal; and

wherein the encrypting allows another device to verify, without decrypting the ciphertext, that the plaintext identifier is not identical to another plaintext identifier maintained by the other device.

48. (Original) One or more computer-readable media as recited in claim 47, wherein generating the ciphertext comprises:

generating, based on the plaintext identifier, a mapped identifier;

encoding the mapped identifier; and

encrypting the encoded identifier.

49. (Original) One or more computer-readable media as recited in claim 48, wherein generating the ciphertext further comprises:

generating, based on the mapped identifier, a decasified identifier and corresponding case information;

wherein the encoding comprises encoding the decasified identifier; and

wherein the encrypting comprises encrypting both the encoded decasified identifier and the case information.

50. (Original) One or more computer-readable media as recited in claim 48, wherein generating the mapped identifier comprises:

checking whether the plaintext identifier is equal to one of a plurality of illegal identifiers;

if the plaintext identifier is not equal to one of the plurality of illegal identifiers, then checking whether the plaintext identifier is equal to one of the plurality of illegal identifiers followed by one or more particular characters;

if the plaintext identifier is not equal to one of the plurality of illegal identifiers followed by one or more particular characters, then using the plaintext identifier as the mapped identifier; and

if the plaintext identifier is equal to one of the plurality of illegal identifiers followed by one or more particular characters, then using as the mapped identifier the plaintext identifier with one of the particular characters removed.

51. (Original) One or more computer-readable media as recited in claim 48, wherein encoding the mapped identifier comprises:

reversing the order of characters in the mapped identifier;

removing, from the reversed identifier, all trailing characters of a particular type;

initializing the encoded identifier with a string of one bits equal in number to a number of trailing characters removed form the reversed identifier followed by a zero bit;

selecting a first character from the reversed identifier;

encoding the first character using a first coding table;

adding, to the encoded identifier, a series of zero bits followed by the encoded first character;

for each additional character in the reversed identifier,

selecting the next character in the reversed identifier,

encoding the next character using a second coding table,

adding, to the encoded identifier, a series of zero bits followed by the encoded next character; and

removing any trailing zero bits and the one bit preceding the trailing zero bits from the encoded identifier.

52. (Original) One or more computer-readable media as recited in claim 51, wherein each coding in the first coding table is the same as a corresponding coding in the second coding table, but the second coding table codes additional characters not coded by the first coding table.

53. (Original) One or more computer-readable media as recited in claim 48, wherein encoding the mapped identifier comprises:

reversing the order of characters in the mapped identifier;

removing, from the reversed identifier, all trailing characters of a particular type;

initializing the encoded identifier with a string of one bits equal in number to a number of trailing characters removed form the reversed identifier followed by a zero bit;

selecting a first character from the reversed identifier;

encoding the first character using a first coding table;

adding, to the encoded identifier, a series of zero bits followed by the encoded first character;

for each additional character in the reversed identifier,

selecting the next character in the reversed identifier,

encoding the next character using one of a plurality of additional coding tables,

adding, to the encoded identifier, a series of zero bits followed by the encoded next character; and

removing any trailing zero bits and the one bit preceding the trailing zero bits from the encoded identifier.

54. (Original) One or more computer-readable media as recited in claim 48, wherein encrypting the encoded identifier comprises using a block cipher to encrypted the encoded identifier.

Claims 55-59 (Canceled).

60. (Original) One or more computer-readable media having stored thereon a plurality of instructions that, when executed by one or more processors of a computer, causes the one or more processors to perform acts including:

receiving a plaintext directory entry;

verifying that the plaintext directory entry is syntactically legal;

encrypting the plaintext directory entry only if the plaintext directory entry is syntactically legal;

communicating the encrypted directory entry to another device; and

wherein the encrypting allows the other device to verify, without decrypting the encrypted directory entry, that the directory entry is not identical to any other directory entry maintained by the other device.

61. (Original) One or more computer-readable media as recited in claim 60, wherein the computer is part of a serverless distributed file system.

62. (Original) One or more computer-readable media as recited in claim 60, wherein the plaintext directory entry comprises a file name.

63. (Original) One or more computer-readable media as recited in claim 60, wherein the plaintext directory entry comprises a directory name.

64. (Original) One or more computer-readable media as recited in claim 60, wherein the plaintext directory entry comprises a folder name.

65. (Canceled).

66. (Canceled).

67. (Original) One or more computer-readable media as recited in claim 60, wherein encrypting the plaintext directory entry comprises:

generating, based on the plaintext directory entry, a mapped identifier;

encoding the mapped identifier; and

encrypting the encoded identifier.

68. (Original) One or more computer-readable media as recited in claim 67, further comprising indicating that the received plaintext directory entry cannot be encrypted if the length of the encoded identifier exceeds a fixed encrypted directory entry size by more than one.

69. (Original) One or more computer-readable media as recited in claim 67, wherein encrypting the plaintext directory entry further comprises:

generating, based on the mapped identifier, a decasified identifier and corresponding case information;

wherein the encoding comprises encoding the decasified identifier; and

wherein the encrypting comprises encrypting both the encoded decasified identifier and the case information.

70. (Original) One or more computer-readable media as recited in claim 67, wherein generating the mapped identifier comprises generating the mapped identifier only if the received plaintext directory entry is syntactically legal.

71. (Original) One or more computer-readable media as recited in claim 67, wherein the encoding comprises encoding the mapped identifier only if the received plaintext directory entry is syntactically legal.

72. (Original) One or more computer-readable media as recited in claim 67, wherein generating the mapped identifier comprises:

checking whether the plaintext directory entry is equal to one of a plurality of illegal identifiers;

if the plaintext directory entry is not equal to one of the plurality of illegal identifiers, then checking whether the plaintext directory entry is equal to one of the plurality of illegal identifiers followed by one or more particular characters;

if the plaintext directory entry is not equal to one of the plurality of illegal identifiers followed by one or more particular characters, then using the plaintext directory entry as the mapped identifier; and

if the plaintext directory entry is equal to one of the plurality of illegal identifiers followed by one or more particular characters, then using as the mapped identifier the plaintext directory entry with one of the particular characters removed.

73. (Original) One or more computer-readable media as recited in claim 72, wherein the particular character comprises an underscore.

74. (Original) One or more computer-readable media as recited in claim 67, wherein encoding the mapped identifier comprises:

reversing the order of characters in the mapped identifier;

removing, from the reversed identifier, all trailing characters of a particular type;

initializing the encoded identifier with a string of one bits equal in number to a number of trailing characters removed form the reversed identifier followed by a zero bit;

selecting a first character from the reversed identifier;

encoding the first character using a first coding table;

adding, to the encoded identifier, a series of zero bits followed by the encoded first character;

for each additional character in the reversed identifier,

selecting the next character in the reversed identifier,

encoding the next character using a second coding table,

adding, to the encoded identifier, a series of zero bits followed by the encoded next character; and

removing any trailing zero bits and the one bit preceding the trailing zero bits from the encoded identifier.

75. (Original) One or more computer-readable media as recited in claim 74, wherein each coding in the first coding table is the same as a corresponding coding in the second coding table, but the second coding table codes additional characters not coded by the first coding table.

76. (Original) One or more computer-readable media as recited in claim 74, wherein the characters of a particular type are the characters that are coded to zero using the first coding table.

77. (Original) One or more computer-readable media as recited in claim 74, wherein the first coding table and the second coding table are Huffman coding tables.

78. (Original) One or more computer-readable media as recited in claim 74, wherein each coding in the first coding table is the same as a corresponding coding in the second coding table, but the second coding table codes additional characters not coded by the first coding table.

79. (Original) One or more computer-readable media as recited in claim 74, wherein for the first character and each additional character, encoding the character only if a set of leading bits of the character are zero, and further comprising adding the character to the encoded identifier if the set of leading bits of the character are not zero.

80. (Original) One or more computer-readable media as recited in claim 67, wherein encrypting the encoded identifier comprises using a block cipher to encrypt the encoded identifier.

81. (Original) One or more computer-readable media as recited in claim 60, wherein the encrypting further comprises generating, using a block cipher, the encrypted directory entry having a fixed size.

Claims 82-86 (Canceled).

87. (Original) A system comprising:

means for verifying that a plaintext directory entry is syntactically legal;

means for encrypting the plaintext directory entry only if the plaintext directory entry is syntactically legal;

means for communicating the encrypted directory entry to another device; and

wherein the encrypting allows the other device to verify, without decrypting the encrypted directory entry, that the directory entry is not identical to any other directory entry maintained by the other device.